UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/735,954	12/15/2003	Frederick S. Kaufmann	11223	3405

7590 07/06/2007

John D. Cowart
Teradata Law IP, WHQ-4W
NCR Corporation
1700 S. Patterson Blvd.
Dayton, OH 45479-0001

EXAMINER	
STACE, BRENT S	
ART UNIT	**PAPER NUMBER**
2161	
MAIL DATE	**DELIVERY MODE**
07/06/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

UNITED STATES PATENT AND TRADEMARK OFFICE

Commissioner for Patents
United States Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450
www.uspto.gov

MAILED

JUL 06 2007

Technology Center 2100

BEFORE THE BOARD OF PATENT APPEALS AND INTERFEERENCES

Application Number: 10/735,954
Filing Date: December 15, 2003
Appellant(s): KAUFMANN ET AL.

Howard L. Speight
For Appellant

EXAMINER'S ANSWER

This is in response to the appeal brief filed 4/2/07 appealing from the Office action mailed 11/3/06.

(1) Real Party in Interest

A statement identifying the real party in interest is contained in the brief.

(2) Related Appeals and Interferences

The examiner is not aware of any related appeals, interferences, or judicial proceedings which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

(3) Status of Claims

The statement of the status of claims contained in the brief is incorrect. A correct statement of the status of the claims is as follows:

The application as originally filed contained 23 claims. Claims 24-26 were added in a Response to a Non-Final Office Action dated August 31, 2006. Claims 1-26 are pending. Claims 1-26 are appealed.

(4) Status of Amendments After Final

The appellant's statement of the status of amendments after final rejection contained in the brief is correct.

(5) Summary of Claimed Subject Matter

The summary of claimed subject matter contained in the brief is correct.

(6) Grounds of Rejection to be Reviewed on Appeal

The appellant's statement of the grounds of rejection to be reviewed on appeal is correct.

(7) Claims Appendix

The copy of the appealed claims contained in the Appendix to the brief is correct.

(8) Evidence Relied Upon

5,930,795	Chen et al.	07-1999

DeWitt, David, et al., "Parallel Database Systems: The Future of High Performance Database Processing" (June 1992) p. 1-26

IBM, "Transition tables" IBM UDB Documentation, (Oct. 31, 2006) p. 1-2

Wikipedia, "Database trigger" Wikipedia, (June 25, 2007) p.1-3

(9) Grounds of Rejection

The following ground(s) of rejection are applicable to the appealed claims:

Claim Rejections - 35 USC § 102

The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

Claims 1-26 are rejected under 35 U.S.C. 102(b) as being anticipated by U.S. Patent No. 5,930,795 (Chen et al.).

As to Claims 1, 4, 7, 9, 11, 14, 17, 19, and 21, Chen teaches "A method for processing a trigger associated with a subject table in a relational database, wherein the trigger defines a triggering statement and one or more triggered actions, the method including:

- determining that a triggering statement of a trigger will execute on a subject table row of a subject table; [Chen, col. 5, lines 50-54, lines 60-67 with Chen, col. 12, lines 25-67 with Chen, col. 13, lines 1-5 with Chen, col. 14, lines 8-50]
- requesting a transition table in response to determining that the triggering statement will execute, the transition table including a transition table row, wherein the transition table row comprises at least one value associated with the subject table row; [Chen, col. 5, lines 50-54, lines 60-67 with Chen, col. 12, lines 25-67 with Chen, col. 13, lines 1-5 with Chen, col. 14, lines 8-50]
- reading the transition table row from the transition table; [Chen, col. 5, lines 50-54, lines 60-67with Chen, col. 12, lines 25-67 with Chen, col. 13, lines 1-5 with Chen, col. 14, lines 8-50]
- identifying a processing unit to receive the transition table row and a triggered action of the trigger based on an association between the identified processing unit and a portion of memory; [Chen, col. 5, lines 50-54, lines 60-67 with Chen, col. 12, lines 25-67 with Chen, col. 13, lines 1-5 with Chen, col. 14, lines 8-50 - i.e., in a parallel processing environment, the different processors process different activities; for a particular job a processor has to be identified to perform the job] and
- transmitting the transition table row and the triggered action to the identified processing unit to be processed" [Chen, col. 5, lines 50-54, lines 60-67 with Chen, col. 12, lines 25-67 with Chen, col. 13, lines 1-5 with Chen, col. 14, lines 8-50 - i.e., in a parallel processing environment, the different processors process

different activities; for a particular job a processor has to be identified to perform the job].

As to Claims 2-3, 5-6, 8, 10, 12-13, 15-16, 18, 20, 22, 23, and 24-26, the limitations of these claims are either addressed or rejected in the claim above or in the explanation in the Examiner's response to the Appellant's remarks.

(10) Response to Argument

Appellant's arguments submitted on 04/02/07 with respect to Claims 1-26 have been considered but are not persuasive.

The Appellant's brief, on page 8, challenges the inherent features of Chen based on the Examiner's Response to Arguments in the Final Office action. It appears that the Appellant did not consider a specific teaching in Chen (the use of a <u>**parallel processing**</u> <u>systems, client/server processing systems, **distributed** systems as a database – outlined in more detail below</u>) when considering the Examiner's Response to Arguments in the Final Office action. As such, the response has been modified below for clarity to further show Chen's inherent teachings/functionality.

Appellant argues that, Chen does not disclose:

(a) "identifying a processing unit to receive the transition table row and a triggered action of the trigger based on an association between the processing unit and a portion of memory" (for Claims 1, 11, and 21)

(b) "instructing a first processing unit, in response to determining that the triggering statement of the trigger will execute, to communicate a transition table row to a second processing unit, wherein the transition table row comprises at least one value associated with the subject table row" (for Claims 4 and 14)

(c) "receiving a triggering statement of a trigger to be executed on a subject table row of a subject table and information identifying a processing unit" (for Claims 7 and 17) and

(d) "receiving a triggered action of a trigger associated with a subject table and information identifying the transition table row" (for Claims 9 and 19).

All of the above arguments point out that an operation (e.g., update, delete, insert etc.) on a subject table row (i.e., a row trigger) cause a trigger to be activated (i.e., a triggering statement is executed) and a transition table, which includes a transition table row (i.e., a table which captures the value(s) that is (are) used to update the row(s) in the subject table when the triggered action is applied to the database) is used to capture the changed row (i.e., the value(s) associated with the subject table row). A processing unit (i.e. a CPU) is identified to receive the transition table row to be processed.

Examiner respectfully disagrees. The following are some simple definitions in the art the may aid in understanding the examiner position:

Transition Table (According to IBM DB2 documentation under the NEW_TABLE AS table-name heading (identified as being a type of transition table in the paragraph

under the Transition tables heading) (this evidence was provided with final Office action)): "Specifies the name of the table, which captures the value that is used to update the rows in the database when the triggered action is applied to the database" [p. 1].

Database Trigger (According to Wikipedia (provided with this response)): "A database trigger is procedural code that is automatically executed in response to certain events on a particular table in a database...There are two classes of triggers, they are either "row triggers" or "statement triggers". With row triggers you can define an action for every row of a table, while statement triggers can occur only once per INSERT, UPDATE, or DELETE statement" [p.1, top].

Parallel and Distributed Database System Architecture (According to DeWitt et al. The Future of High Performance Database Processing, 1992 (this evidence was provided with final Office action)): "This architecture is based on a shared-nothing hardware design in which processors communicate with one another only by sending messages via an interconnection network. In such systems, tuples of each relation in the database are partitioned (declustered) across disk storage units attached directly to each processor...Such architectures were pioneered by Teradata in the late seventies and by several research projects. This design is now used in Teradata, Tandem, NCR, Oracle-nCUBE, and several other products currently under development. [p. 3, parts of 2nd full paragraph]...In a shared disk multiprocessor system, each processor has a private memory, but has direct access to all disks" [p. 6, next to shared-disks].

Chen teaches:

> "The system, method and program of this invention is applicable to any type of database management system whether it is contained within a single system or is within a networked environment including **parallel processing** systems, client/server processing systems, **distributed** systems, etc. Although the invention herein is described in reference to relational database management systems … adaptable to other database management systems … the system, method and program of this invention is also applicable to triggers. For example, it has been proposed that in a future release of the IBM DB2 Common Server that the standard trigger declaration be extended to allow the access of transition tables in the trigger body written in external host languages as well as the SQL language" [Chen, col. 5, lines 49-67] (emphasis added).
>
> "CREATE TRIGGER fuh.mytrig
>
> ```
> "CREATE TRIGGER fuh.mytrig
>         AFTER UPDATE OF c1, c2 ON fuh.mytbl
>         REFERENCE NEW_TABLE AS nt
>                 OLD_TABLE AS ot
>         FOR EACH STATEMENT
>         MODE DB2SQL
> BEGIN
> ...
> END
> ```
>
> Transition tables are created and populated in the executable plan for triggering SQL operation. Let the table fuh.mytbl consist of five rows, {(1,100), (2,10), (3, -1001), (4,331), (5, -5)}, and fuh.mytrig be the trigger as shown in the previous example" [Chen, col. 12, lines 24-38].
>
> "UPDATE fuh.mytbl SET c2=-c2 WHERE c2<0 will render the transition tables nt and ot to be created and populated with the following rows. nt={(3,1001), (5,5)} ot= {(3, -1001), (5, -5)}" [Chen, col. 12, lines 47-52].

The preceding text excerpts clearly indicate that a query setting the c2 to –c2, where c2<0 in the subject table fuh.mytbl would activate the trigger fuh.mytrig and, within the trigger, transition table nt (new table) would be populated by the affected rows of the

subject table (i.e., values associated with the subject table row) ("ot" (old table) in the reference is the subject/base table). The transition table row needs to be written to the actual table after the trigger is executed. Now, in a distributed parallel database system (the Chen method can be implemented in such a system as cited above [Chen, col. 5, lines 49-67]), the associated actual affected data corresponding to the new data is stored in a particular disk storage/portion of memory (and has an associated second processor/CPU). Therefore, in order to execute properly, the database management system identifies that particular second processor, since it has the data, and sends the transition table data to that processor to be processed. Such characteristics is fundamental to any functioning distributed parallel database system e.g., Teradata, IBM DB2, Oracle etc. Examiner is not sure what the Appellant is claiming since the Appellant appears to be claiming a fundamental characteristic of a database system, which has existed for decades (according to Dewitt since seventies). Examiner sincerely believes the above explanation answers all of the arguments raised by the Appellant. All of those arguments are essentially same thing in varying terminologies.

According to the definitions above, a distributed parallel database management system (e.g., Oracle, Teradata, IBM DB2, Chen etc.) executing a triggered statement using a transition table to capture the value that is used to update the subject table row in the database when the triggered action is applied to the database. The table(s) and/or row(s) are partitioned across multiple disks, wherein the disk(s) are attached directly to the processors. For understanding purposes, the examiner's position is

outlined here in an example. Since Chen discloses that Chen's invention works on a distributed parallel database management system (see above cited section), an exemplary table AB row is divided into disk A (and attached processor A) and disk B (and attached processor B) (as justified by the definitions above, specifically, Parallel and Distributed Database System Architecture) and an exemplary UPDATE on the table AB row triggers an event, the transition table captures the new values and a processor has to be identified (i.e., a disk/memory portion and the associated processor has to be identified, which stores the relevant portion of the table row that needs to be updated) to process the UPDATE (when a disk does not have the portion of the table that needs to be updated there is no point in going to that disk (evidenced by the distributed/paralleled aspects of the database management system under consideration) – however it will go to the disk/processor that does have the portion of the table that needs to be updated since that disk/processor requires/does updating). All of the "receiving triggered statement", "determining triggered statement", "computer program", "executing triggered statement" are all inherent characteristics of any database management system supporting triggers. Without receiving, determining and executing, a triggered statement cannot be processed. Identifying a processor to process an event (e.g., UPDATE, DELETE etc.) is inherent to a distributed parallel database system, where data is distributed/partitioned among different portions of memory (e.g., different disks), because simply a processor must be used to process commands/events and there is no need to access other portions of memory/processors when the other memory portions do not have the corresponding data that needs to be processed. Identifying a

processor is done at the database management system level and not at the user level. This is evidenced by the functioning invention of Chen as a whole especially on a distributed parallel database management system.

Any other arguments by the Appellant are more limiting or irrelevant than the claimed language.

Additionally, other claims argued merely because of a dependency on a previously argued claim(s) or because they are substantially the same as a previously argued claim(s) in the brief presented to the examiner, filed April 2nd 20007, are moot in view of the examiner's interpretation of the claims and art and are still considered rejected based on their respective rejections from a prior Office action (part(s) of recited again above).

(11) Related Proceeding(s) Appendix

No decision rendered by a court or the Board is identified by the examiner in the Related Appeals and Interferences section of this examiner's answer.

For the above reasons, it is believed that the rejections should be sustained.

Respectfully submitted,

Brent Stace Brent Stace

Conferees:

Apu M. Mofiz

APU MOFIZ
SUPERVISORY PATENT EXAMINER

Eddie Lee

EDDIE C. LEE
SUPERVISORY PATENT EXAMINER

Database trigger

From Wikipedia, the free encyclopedia
(Redirected from Database Trigger)

A **database trigger** is procedural code that is automatically executed in response to certain events on a particular table in a database. Triggers can restrict access to specific data, perform logging, or audit data modifications.

There are two classes of triggers, they are either "row triggers" or "statement triggers". With row triggers you can define an action for every row of a table, while statement triggers occur only once per INSERT, UPDATE, or DELETE statement. Triggers cannot be used to audit data retrieval via SELECT statements.

Each class can be of several types. There are "BEFORE triggers" and "AFTER triggers" which identifies the time of execution of the trigger. There is also an "INSTEAD OF trigger" which is a trigger that will execute instead of the triggering statement.

There are typically three triggering events that cause triggers to 'fire':

- INSERT event (as a new record is being inserted into the database).
- UPDATE event (as a record is being changed).
- DELETE event (as a record is being deleted).

The major features and effects of database triggers are that they:

- do not accept parameters or arguments
- cannot perform commit or rollback operations because they are part of the triggering SQL statement (only through autonomous transactions)
- can cause mutating table errors, if they are poorly written.

Contents

Triggers in Oracle

In addition to triggers that fire when data is modified, Oracle 9i supports triggers that fire when schema objects (that is, tables) are modified and when user logon or logoff events occur. These trigger types are referred to as "Schema-level triggers".

Schema-level triggers

- Before Creation
- After Creation

- Before Alter
- After Alter
- Before Drop
- After Drop
- Before Logoff
- After Logon

Triggers in Microsoft SQL Server

Microsoft SQL Server supports triggers either after or instead of an insert, update, or delete operation.

Microsoft SQL Server supports triggers on tables and views with the constraint that a view can be referenced only by an INSTEAD OF trigger.

Microsoft SQL Server 2005 introduced support for Data Definition Language (DDL) triggers, which can fire in reaction to a very wide range of events, including:

- Drop table
- Create table
- Alter table
- Login events

A full list (http://msdn2.microsoft.com/en-us/library/ms189871(SQL.90).aspx) is available on MSDN.

Triggers in PostgreSQL

PostgreSQL introduced support for triggers in 1997. The following functionality in SQL:2003 is not implemented in PostgreSQL:

- SQL allows triggers to fire on updates to specific columns; PostgreSQL does not support this feature.
- The standard allows the execution of a number of other SQL statements than SELECT, INSERT, UPDATE, such as CREATE TABLE as the triggered action.

Triggers in MySQL

MySQL 5.0 introduced support for triggers. Some of the triggers MYSQL supports are

- INSERT Trigger
- UPDATE Trigger
- DELETE Trigger

SQL:2003 requirements

The SQL:2003 standard mandates that triggers give programmers access to record variables by means of a syntax such as *REFERENCING NEW AS n*. For example, if a trigger is monitoring for changes to a salary column one could write a trigger like the following:

```
CREATE TRIGGER salary_trigger
   BEFORE UPDATE ON employee_table
   REFERENCING NEW ROW AS n, OLD ROW AS o
   FOR EACH ROW
   IF n.salary <> o.salary THEN
      do whatever chages you want to perform;
   END IF;
```

External links

- Microsoft SQL Server CREATE TRIGGER (http://msdn2.microsoft.com/en-us/library/aa258254(SQL.80).aspx)
- Microsoft SQL Server DROP TRIGGER (http://msdn2.microsoft.com/en-us/library/aa258846(SQL.80).aspx)
- MySQL Database triggers(http://dev.mysql.com/doc/refman/5.0/en/triggers.html)
- DB2 CREATE TRIGGER statement (http://publib.boulder.ibm.com/infocenter/db2luw/v9/topic/com.ibm.db2.udb.admin.doc/doc/r0000931.htm)
- PostgreSQL CREATE TRIGGER (http://www.postgresql.org/docs/8.2/static/sql-createtrigger.html)

Topics in database management systems (DBMS) ( )

Concepts
Database • Database models • Database storage • Relational model • Distributed DBMS • ACID • Null
Relational database • Relational algebra • Relational calculus • Database normalization • Referential integrity • Relational DBMS
Primary key, Foreign key, Surrogate key, Superkey, Candidate key

Objects
Trigger • View • Table • Cursor • Log • Transaction • Index
Stored procedure • Partition

Topics in SQL
Select • Insert • Update • Merge • Delete • Join • Union • Create • Drop
Begin work • Commit • Rollback • Truncate • Alter

Implementations of database management systems

Types of implementations
Relational • Flat file • Deductive • Dimensional • Hierarchical • Object oriented • Object relational • Temporal • XML data stores

Database products
Object-oriented (comparison) • Relational (comparison)

Components
Query language • Query optimizer • Query plan • ODBC • JDBC

Category: Database management systems